

**Laporan Tugas Kecil 1**  
**IF2211 Strategi Algoritma**

**Semester II tahun 2022/2023**

**Penyelesaian Permainan Kartu 24**  
**dengan Algoritma Brute Force**



Disusun oleh

Jauza Lathifah Annassalafi

13521030

**Program Studi Teknik Informatika**  
**Sekolah Teknik Elektro dan Informatika**  
**Institut Teknologi Bandung**  
**2022**

# DAFTAR ISI

<b>DAFTAR ISI</b>	<b>2</b>
<b>BAB 1</b>	<b>4</b>
<b>ALGORITMA BRUTE FORCE</b>	<b>4</b>
1.1 Permainan Kartu 24	4
1.2 Algoritma Brute Force	4
1.3 Alur Kerja Program	5
<b>BAB 2</b>	<b>7</b>
<b>PROGRAM</b>	<b>7</b>
2.1 Library	7
2.2 Fungsi	7
2.2.1 void intro()	7
2.2.2 float convertCardToFloat(string card)	8
2.2.3 string convertCardToString(int card)	9
2.2.4 void intToStr(int num, string &str)	9
2.2.5 bool isCardValid(string card)	10
2.2.6 void randomCard()	11
2.2.7 void cardInput()	11
2.2.8 float operation(float cardA, float cardB, int sign)	12
2.2.9 string sign(int op)	13
2.2.10 bool checkDouble (vector<string> hasil, string temp)	14
2.2.11 void check24(float card1, float card2, float card3, float card4, vector<string> *hasil)	14
2.2.12 void cardPosition(float card1, float card2, float card3, float card4, vector<string> *hasil)	15
2.2.13 void printSolution(vector<string> hasil)	16
2.2.14 void saveFile(vector<string> hasil)	17
2.3. Program Utama	18
<b>BAB 3</b>	<b>20</b>
<b>TEST CASE</b>	<b>20</b>
3.1 Test Case 1 (Random)	20
3.2 Test Case 2 (Random)	22
3.3 Test Case 3 (Random)	24
3.4 Test Case 4 (Keyboard)	26
3.5 Test Case 5 (Keyboard)	27
3.6 Test Case 6 (Keyboard)	30
3.7 Test Case 7 (Keyboard)	31

3.8 Test case 8 (Keyboard)	32
3.9 Test Case 9 (Keyboard)	33
<b>BAB 4</b>	<b>35</b>
<b>LAMPIRAN</b>	<b>35</b>
4.1 Link Repository	35
4.2 Cek List	35

# BAB 1

## ALGORITMA BRUTE FORCE

### 1.1 Permainan Kartu 24

Cara bermainnya adalah mencari cara untuk mengubah 4 buah angka random dengan melakukan operasi-operasi perhitungan sehingga mendapatkan hasil akhir sejumlah 24. Permainan Kartu 24 biasa dimainkan dengan menggunakan kartu remi yang terdiri dari 52 kartu. Kartu remi terbagi menjadi empat suit (sekop, hati, keriting, dan wajik) yang masing-masing terdiri dari 13 kartu (As, 2, 3, 4, 5, 6, 7, 8, 9, 10, Jack, Queen, dan King). Yang perlu diperhatikan hanyalah nilai kartu yang didapat (As, 2, 3, 4, 5, 6, 7, 8, 9, 10, Jack, Queen, dan King). As bernilai 1, Jack bernilai 11, Queen bernilai 12, King bernilai 13, sedangkan kartu bilangan memiliki nilai dari bilangan itu sendiri. Pertama, ambil 4 kartu. Kedua, lakukan perhitungan dengan keempat kartu tersebut menggunakan operasi dasar matematika penjumlahan (+), pengurangan (-), perkalian ( $\times$ ), divisi (/) dan tanda kurung ( ) dan tiap kartu harus digunakan tepat sekali dan urutan penggunaannya bebas. Permainan berakhir ketika pemain berhasil menemukan solusi untuk membuat kumpulan nilainya menjadi 24.

Pada tugas kecil 1 ini, diberi tugas untuk membuat program sederhana dalam Bahasa C++ yang mengimplementasikan algoritma Brute Force untuk mencari seluruh solusi permainan kartu 24.

### 1.2 Algoritma Brute Force

Brute force adalah sebuah pendekatan yang lempang (straight forward) untuk memecahkan suatu masalah, biasanya didasarkan pada pernyataan masalah (problem statement) dan definisi konsep yang dilibatkan. Algoritma brute force memecahkan masalah dengan sangat sederhana, langsung dan dengan cara yang jelas. Pada dasarnya, dengan algoritma brute force akan diperiksa semua kemungkinan yang ada untuk memberikan solusi dari permasalahan. Algoritma brute force umumnya tidak cerdas, karena ia membutuhkan jumlah langkah yang besar dalam penyelesaiannya. Algoritma brute force seringkali merupakan pilihan yang kurang disukai karena ketidakmangkusannya itu, tetapi dengan mencari pola-pola yang mendasar, keteraturan, atau trik-trik khusus, biasanya akan membantu kita menemukan algoritma yang lebih cerdas dan lebih mangkus.

Pada permainan 24, terdapat banyak kemungkinan untuk mendapatkan hasil 24 dari 4 kartu dengan 4 operator dan 4 penempatan tanda kurung dalam operasi perhitungan. Didapatkan jumlah seluruh kemungkinan solusi dari program yang dibuat

adalah  $24 \times 64 \times 4 = 6.144$  buah kemungkinan solusi. Angka-angka tersebut didapat sebagai berikut:

1. Susunan keempat kartu/angka dalam perhitungan  
Banyaknya kartu/angka yang digunakan dalam perhitungan adalah 4 sehingga permutasinya adalah  $4! = 24$  kemungkinan susunan kartu/angka dalam perhitungan untuk menghasilkan 24.
2. Susunan operator dalam perhitungan  
Banyak operator yang digunakan adalah 4 buah, yaitu tambah, kurang, kali, dan bagi. Pada permainan ini digunakan 4 kartu/angka sehingga terdapat 3 posisi operator untuk melakukan perhitungan. Oleh karena itu, banyaknya kemungkinan susunan operator untuk menghasilkan angka 24 adalah  $4^3$  (4 operator dengan 3 posisi).
3. Penempatan tanda kurung dalam perhitungan  
Penempatan tanda kurung sangat mempengaruhi hasil sebuah perhitungan karena memiliki arti untuk menentukan urutan operasi mana yang akan dikerjakan terlebih dahulu. Sebagai contoh dengan 4 buah kartu/angka yang dimisalkan dengan A, B, C, dan D dan sebuah operasi yang dimisalkan dengan @, maka susunan penempatan tanda kurungnya adalah:
  1.  $((A @ B) @ C) @ D$
  2.  $(A @ B) @ (C @ D)$
  3.  $(A @ (B @ C)) @ D$
  4.  $(A @ ((B @ C) @ D))$

### 1.3 Alur Kerja Program

Program bekerja dengan alur sebagai berikut:

1. Pada saat program dijalankan, program akan memberikan pilihan untuk mendapatkan 4 kartu, yaitu input angka 1 untuk memasukkan kartu/angka dari keyboard dan input angka 2 untuk menghasilkan kartu/angka secara acak. Jika memasukan sebuah input yang tidak sesuai, maka program akan terus meminta sebuah inputan hingga benar. Ketika memilih untuk memasukkan kartu/kartu dari keyboard, haruslah memasukkan 4 kartu/angka, yaitu angka 2-10 atau huruf A, J, Q, dan K yang dipisahkan dengan spasi atau enter. Program akan terus kembali meminta inputan 4 kartu/ angka jika melakukan input selain dari angka 2-10 atau huruf A, J, Q, dan K.
2. Setelah mendapatkan 4 kartu/angka, program akan menampilkan solusi-solusi dari keempat kartu/angka tersebut dan waktu eksekusi dalam mendapatkan solusi dari 4 kartu/angka tersebut. Solusi-solusi tersebut didapatkan dengan cara:

- 1) Fungsi `cardPosition()` yang dipanggil di program utama akan membuat kemungkinan susunan terhadap 4 kartu/angka, yaitu sebanyak  $4! = 24$ .
  - 2) Keempat kartu/angka akan dimasukkan ke dalam fungsi `check24()` dengan susunan kartu yang berbeda-beda, yaitu sebanyak 24. Fungsi `check24()` digunakan untuk membuat setiap kemungkinan 4 operator aritmatika yang akan ditempatkan di antara susunan 4 kartu/angka dan juga 4 penempatan tanda kurung yang berbeda hingga mendapatkan hasil 24. Jika hasil aritmatika adalah 24 maka susunan 4 kartu/angka dengan operator dan penempatan tanda kurungnya akan diubah ke dalam bentuk string lalu dimasukkan ke dalam sebuah vektor. Apabila string tersebut sudah terdapat dalam vektor, maka tidak akan dimasukkan lagi ke dalam vektor, sehingga tidak terdapat solusi yang sama.
  - 3) Banyaknya solusi dan solusi-solusi dalam bentuk string yang terdapat pada vektor akan ditampilkan dengan memanggil fungsi `printSolution()`.
3. Program juga menghitung waktu eksekusi program dan memberikan pilihan untuk menyimpan solusi-solusi tersebut kedalam sebuah file.

Gambar 2.2.1 intro

### 2.2.2 float convertCardToFloat(string card)

Fungsi ini akan dipanggil pada fungsi randomCard dan cardInput yang digunakan sebagai mengubah inputan yang bertipe data string menjadi tipe data float/double sehingga dapat diproses dalam perhitungan.

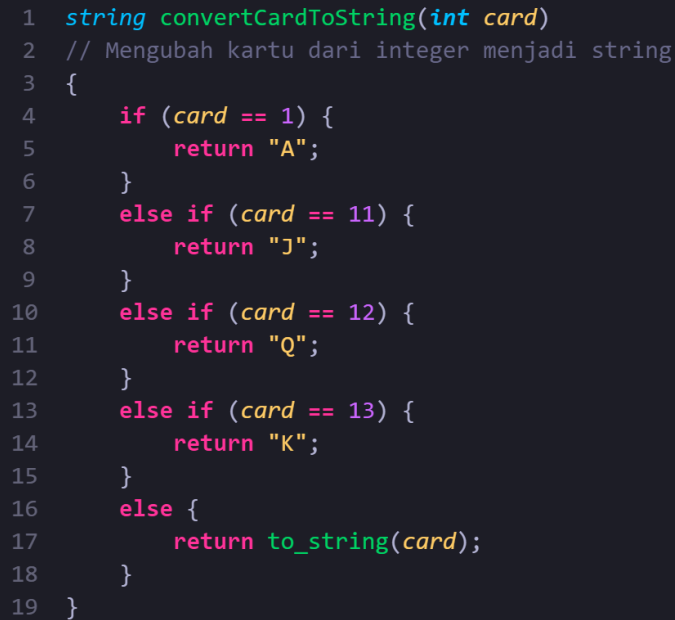
```
1 float convertCardToFloat(string card)
2 // Mengubah kartu dari string menjadi float
3 {
4     if (card == "A") {
5         return 1;
6     }
7     else if (card == "J") {
8         return 11;
9     }
10    else if (card == "Q") {
11        return 12;
12    }
13    else if (card == "K") {
14        return 13;
15    }
16    else if (card == "2") {
17        return 2;
18    }
19    else if (card == "3") {
20        return 3;
21    }
22    else if (card == "4") {
23        return 4;
24    }
25    else if (card == "5") {
26        return 5;
27    }
28    else if (card == "6") {
29        return 6;
30    }
31    else if (card == "7") {
32        return 7;
33    }
34    else if (card == "8") {
35        return 8;
36    }
37    else if (card == "9") {
38        return 9;
39    }
40    else if (card == "10") {
41        return 10;
42    }
43    else {
44        return 0;
45    }
46 }
```

Gambar 2.2.2 convertCardToFloat



### 2.2.3 string convertCardToString(int card)

Fungsi ini akan dipanggil pada fungsi randomCard dalam mengacak keluaran kartu yang berfungsi untuk mengubah tipe data angka yang dihasilkan oleh program random, yaitu berupa tipe data integer dan kemudian akan diubah menjadi string oleh fungsi convertCardToString.



```
1  string convertCardToString(int card)
2  // Mengubah kartu dari integer menjadi string
3  {
4      if (card == 1) {
5          return "A";
6      }
7      else if (card == 11) {
8          return "J";
9      }
10     else if (card == 12) {
11         return "Q";
12     }
13     else if (card == 13) {
14         return "K";
15     }
16     else {
17         return to_string(card);
18     }
19 }
```

Gambar 2.2.3 convertCardToString

### 2.2.4 void intToStr(int num, string &str)

Fungsi ini akan dipanggil pada randomCard yang berfungsi untuk mengubah angka yang terpilih dari program random menjadi bentuk string. Apabila terpilih angka 1, 11, 12, atau 13 akan diubah menjadi A, J, Q, atau K. Berbeda dengan fungsi sebelumnya convertCardToString yang menghasilkan nilai balik (return values) sedangkan intToStr tidak menghasilkan nilai balik.

```

1 void intToStr(int num, string &str)
2 // Mengubah int menjadi string
3 {
4     if (num == 1) {
5         str = "A";
6     }
7     else if (num == 11) {
8         str = "J";
9     }
10    else if (num == 12) {
11        str = "Q";
12    }
13    else if (num == 13) {
14        str = "K";
15    }
16    else {
17        str = to_string(num);
18    }
19 }

```

Gambar 2.2.4 IntToStr

### 2.2.5 bool isCardValid(string card)

Fungsi ini akan dipanggil pada saat menginput kartu dengan keyboard atau dalam fungsi cardInput yang berfungsi untuk mengecek kartu yang diinput apakah sudah sesuai dengan ketentuan (angka 2-10 atau huruf A, J, Q, dan K). Jika sudah sesuai akan mengembalikan nilai "true", apabila tidak akan mengembalikan nilai "false".

```

1 bool isCardValid(string card)
2 // Mengecek apakah kartu valid
3 {
4     if (card == "A" || card == "J" || card == "Q" ||
5     card == "K" || card == "2" || card == "3" ||
6     card == "4" || card == "5" || card == "6" ||
7     card == "7" || card == "8" || card == "9" ||
8     card == "10") {
9         return true;
10    }
11    else {
12        return false;
13    }
14 }

```

Gambar 2.2.5 isCardValid

### 2.2.6 void randomCard()

Fungsi ini akan dipanggil di cardInput yang berfungsi untuk mengeluarkan 4 kartu secara acak.



```
1 void randomCard()
2 // Membuat 4 kartu secara acak
3 {
4     int c1, c2, c3, c4;
5     srand(time(NULL));
6     c1 = rand() % 13 + 1;
7     c2 = rand() % 13 + 1;
8     c3 = rand() % 13 + 1;
9     c4 = rand() % 13 + 1;
10    intToStr(c1, input1);
11    intToStr(c2, input2);
12    intToStr(c3, input3);
13    intToStr(c4, input4);
14    card1 = convertCardToFloat(convertCardToString(c1));
15    card2 = convertCardToFloat(convertCardToString(c2));
16    card3 = convertCardToFloat(convertCardToString(c3));
17    card4 = convertCardToFloat(convertCardToString(c4));
18    cout << " " << endl;
19    cout << " ----- " << endl;
20    cout << "\t" << "Kartu yang terpilih adalah " << input1 <<
21    ", " << input2 << ", " << input3 << ", dan " << input4 << "
22    \t\t\t\t\t" << endl;
23 }
24 }
```

Gambar 2.2.6 randomCard

### 2.2.7 void cardInput()

Fungsi ini akan dipanggil di program utama yang berfungsi untuk menerima inputan 4 kartu dari keyboard dan secara acak oleh program (randomCard). Di bagian ini terdapat validasi inputan kartu. Jika kartu yang dimasukkan tidak sesuai dengan ketentuan (angka 2-10 atau huruf A, J, Q, dan K) maka program akan meminta kembali 4 kartu hingga valid dengan memanggil fungsi iscardValid.

```

1 void cardInput()
2 // Membaca keempat kartu untuk permainan 24
3 {
4     string pilihan;
5
6     //pilih input dari keyboard atau random
7     cout << " " << endl;
8     cout << " \t" << "Pilih input dari keyboard atau random:" << endl;
9     cout << " \t" << "1. Keyboard" << endl;
10    cout << " \t" << "2. Random" << endl;
11    cout << " " << endl;
12    cout << " \t" << "Masukkan pilihan : ";
13    cin >> pilihan;
14
15    // Mengecek apakah input valid
16    while (pilihan != "1" && pilihan != "2") {
17        cout << " " << endl;
18        cout << " \t" << "Input tidak valid. Masukkan pilihan (1/2): ";
19        cin >> pilihan;
20    }
21
22    // Input dari keyboard
23    if (pilihan == "1") {
24        cout << " " << endl;
25        cout << " \t" << "Masukkan 4 kartu: " << endl;
26        cout << " \t" << "(angka 2-10 atau huruf A, J, Q, dan K)" << endl;
27        cout << " " << endl;
28        cout << " \t" << "Masukkan kartu : ";
29        cin >> input1 >> input2 >> input3 >> input4;
30        card1 = convertCardToFloat(input1);
31        card2 = convertCardToFloat(input2);
32        card3 = convertCardToFloat(input3);
33        card4 = convertCardToFloat(input4);
34
35        // Mengecek apakah input kartu valid
36        while (!isCardValid(input1) || !isCardValid(input2) ||
37              !isCardValid(input3) || !isCardValid(input4)) {
38            cout << " " << endl;
39            cout << " \t" << "Kartu tidak valid. Masukkan kembali 4 kartu. " << endl;
40            cout << " \t" << "(angka 2-10 atau huruf A, J, Q, dan K)" << endl;
41            cout << " " << endl;
42            cout << " \t" << "Masukkan 4 kartu: ";
43            cin >> input1 >> input2 >> input3 >> input4;
44            card1 = convertCardToFloat(input1);
45            card2 = convertCardToFloat(input2);
46            card3 = convertCardToFloat(input3);
47            card4 = convertCardToFloat(input4);
48        }
49        cout << " " << endl;
50        cout << " -----
51        ----- " << endl;
52        cout << " \t" << "Kartu yang anda pilih adalah " << input1 << ", " << input2 <<
53        ", " << input3 << ", dan " << input4 << " \t \t \t \t \t " << endl;
54    }
55    // Input random
56    else if (pilihan == "2") {
57        randomCard();
58    }
59 }

```

Gambar 2.2.7 cardInput

## 2.2.8 float operation(float cardA, float cardB, int sign)

Fungsi ini akan dipanggil di check24 yang berfungsi melakukan perhitungan pertambahan, pengurangan, perkalian, ataupun pembagian terhadap 2 angka/kartu

```

1 float operation(float cardA, float cardB, int sign)
2 // Menghitung operasi cardA dan cardB
3 {
4     if (sign == 1) {
5         return cardA + cardB;
6     }
7     else if (sign == 2) {
8         return cardA - cardB;
9     }
10    else if (sign == 3) {
11        return cardA * cardB;
12    }
13    else if (sign == 4) {
14        return cardA / cardB;
15    }
16    else {
17        return 0;
18    }
19 }

```

Gambar 2.2.8 operation

### 2.2.9 string sign(int op)

Fungsi ini akan dipanggil pada check24 yang berfungsi untuk mengubah angka menjadi sebuah operator penjumlahan, pengurangan, perkalian, ataupun pembagian.

```

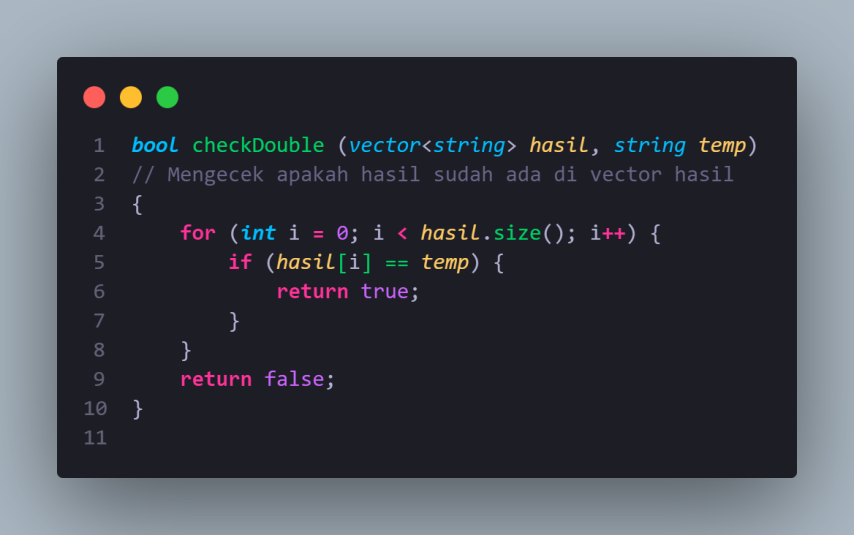
1 string sign(int op)
2 // Mengubah angka menjadi operator
3 {
4     if (op == 1) {
5         return "+";
6     }
7     else if (op == 2) {
8         return "-";
9     }
10    else if (op == 3) {
11        return "*";
12    }
13    else if (op == 4) {
14        return "/";
15    }
16    else {
17        return 0;
18    }
19 }

```

Gamvar 2.2.9 sign

### 2.2.10 bool checkDouble (vector<string> hasil, string temp)

Fungsi ini akan dipanggil pada check24 yang berfungsi untuk mengecek apakah solusi yang didapatkan sudah terdapat pada vektor sehingga tidak ada solusi yang sama.



```
1  bool checkDouble (vector<string> hasil, string temp)
2  // Mengecek apakah hasil sudah ada di vector hasil
3  {
4      for (int i = 0; i < hasil.size(); i++) {
5          if (hasil[i] == temp) {
6              return true;
7          }
8      }
9      return false;
10 }
11
```

Gambar 2.2.10 checkDouble

### 2.2.11 void check24(float card1, float card2, float card3, float card4, vector<string> \*hasil)

Fungsi ini akan dipanggil pada program utama yang berfungsi untuk mengecek apakah suatu perhitungan yang terdiri dari 4 kartu, 4 operator, dan penempatan tanda kurung mengeluarkan hasil 24 yang selanjutnya akan dimasukkan kedalam vector. Akan dipanggil fungsi checkDouble sehingga tidak terdapat solusi yang sama.

```

1 void check24(float card1, float card2, float card3, float card4, vector<string> *hasil)
2 // Mengecek apakah ada kombinasi kartu yang hasil perhitungannya 24
3 {
4     int op1, op2, op3;
5     float calculate;
6     string result = "";
7
8     for (op1 = 1; op1 <= 4; op1++) {
9         for (op2 = 1; op2 <= 4; op2++) {
10             for (op3 = 1; op3 <= 4; op3++) {
11                 // ((card1 op1 card2) op2 card3) op3 card4
12                 calculate = operation(operation(card1, card2, op1), card3, op2), card4, op3);
13                 if (calculate == 24) {
14                     result = "(" + to_string((int)card1) + " " + sign(op1) + " " + to_string((int)card2) +
15                         " " + sign(op2) + " " + to_string((int)card3) + " " + sign(op3) + " " + to_string((int)card4) +
16                         " = " + to_string((int)calculate);
17                     if (!checkDouble(*hasil, result)) {
18                         (*hasil).push_back(result);
19                         solusi++;
20                     }
21                 }
22                 // (card1 op1 card2) op2 (card3 op3 card4)
23                 calculate = operation(operation(card1, card2, op1), operation(card3, card4, op3), op2);
24                 if (calculate == 24) {
25                     result = "(" + to_string((int)card1) + " " + sign(op1) + " " + to_string((int)card2) + " " +
26                         sign(op2) + " (" + to_string((int)card3) + sign(op3) + " " + to_string((int)card4) + " = " +
27                         to_string((int)calculate);
28                     if (!checkDouble(*hasil, result)) {
29                         (*hasil).push_back(result);
30                         solusi++;
31                     }
32                 }
33                 // card1 op1 ((card2 op2 card3) op3 card4)
34                 calculate = operation(card1, operation(operation(card2, card3, op2), card4, op3), op1);
35                 if (calculate == 24) {
36                     result = to_string((int)card1) + " " + sign(op1) + " (" + to_string((int)card2) + sign(op2) +
37                         " " + to_string((int)card3) + " " + sign(op3) + " " + to_string((int)card4) + " = " +
38                         to_string((int)calculate);
39                     if (!checkDouble(*hasil, result)) {
40                         (*hasil).push_back(result);
41                         solusi++;
42                     }
43                 }
44                 // (card1 op1 (card2 op2 card3)) op3 card4
45                 calculate = operation(operation(card1, operation(card2, card3, op2), op1), card4, op3);
46                 if (calculate == 24) {
47                     result = "(" + to_string((int)card1) + " " + sign(op1) + " (" + to_string((int)card2) + sign(op2) +
48                         " " + to_string((int)card3) + " " + sign(op3) + " " + to_string((int)card4) + " = " +
49                         to_string((int)calculate);
50                     if (!checkDouble(*hasil, result)) {
51                         (*hasil).push_back(result);
52                         solusi++;
53                     }
54                 }
55             }
56         }
57     }
58 }
59

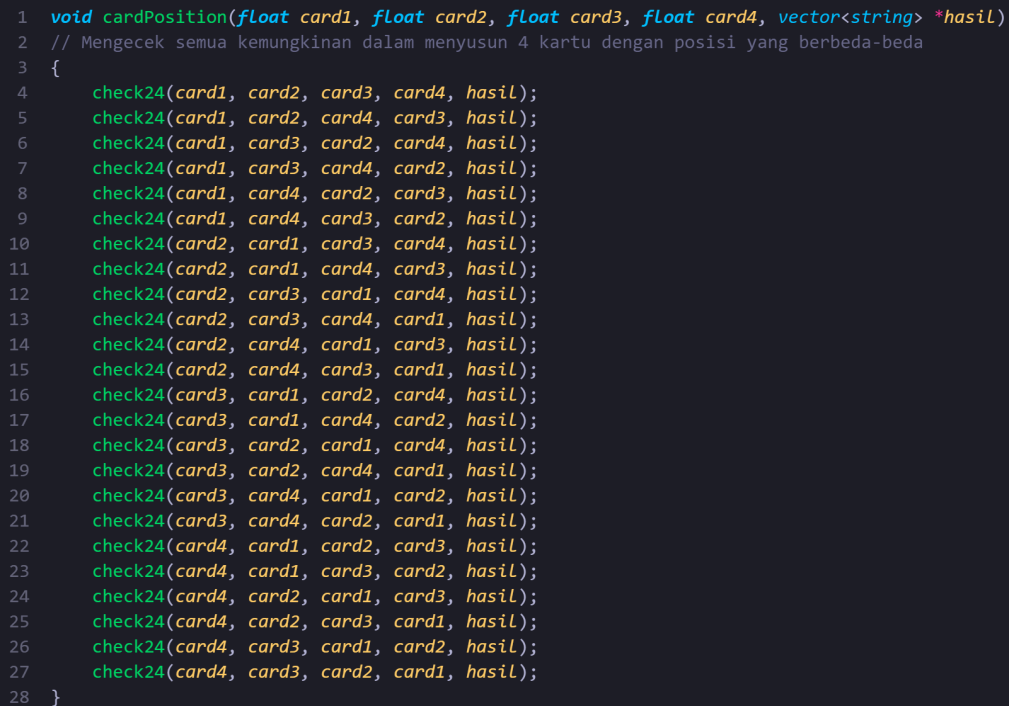
```

Gambar 2.2.11 check24

## 2.2.12 void cardPosition(float card1, float card2, float card3, float card4, vector<string> \*hasil)

Fungsi ini memanggil fungsi check24() sebanyak 24 kali dengan susunan kartu/angka yang berbeda-beda untuk mengecek semua kemungkinan dalam

menyusun 4 kartu dengan posisi yang berbeda-beda dan akan dipanggil di program utama.

A screenshot of a code editor with a dark background and light-colored text. The code is written in C++ and defines a function named `cardPosition`. The function takes four float parameters (`card1`, `card2`, `card3`, `card4`) and a vector of strings (`hasil`) by reference. It contains a comment in Indonesian: `// Mengecek semua kemungkinan dalam menyusun 4 kartu dengan posisi yang berbeda-beda`. The function body consists of a loop that calls `check24` with all possible permutations of the four cards. The code is numbered from 1 to 28.

```
1 void cardPosition(float card1, float card2, float card3, float card4, vector<string> *hasil)
2 // Mengecek semua kemungkinan dalam menyusun 4 kartu dengan posisi yang berbeda-beda
3 {
4     check24(card1, card2, card3, card4, hasil);
5     check24(card1, card2, card4, card3, hasil);
6     check24(card1, card3, card2, card4, hasil);
7     check24(card1, card3, card4, card2, hasil);
8     check24(card1, card4, card2, card3, hasil);
9     check24(card1, card4, card3, card2, hasil);
10    check24(card2, card1, card3, card4, hasil);
11    check24(card2, card1, card4, card3, hasil);
12    check24(card2, card3, card1, card4, hasil);
13    check24(card2, card3, card4, card1, hasil);
14    check24(card2, card4, card1, card3, hasil);
15    check24(card2, card4, card3, card1, hasil);
16    check24(card3, card1, card2, card4, hasil);
17    check24(card3, card1, card4, card2, hasil);
18    check24(card3, card2, card1, card4, hasil);
19    check24(card3, card2, card4, card1, hasil);
20    check24(card3, card4, card1, card2, hasil);
21    check24(card3, card4, card2, card1, hasil);
22    check24(card4, card1, card2, card3, hasil);
23    check24(card4, card1, card3, card2, hasil);
24    check24(card4, card2, card1, card3, hasil);
25    check24(card4, card2, card3, card1, hasil);
26    check24(card4, card3, card1, card2, hasil);
27    check24(card4, card3, card2, card1, hasil);
28 }
```

Gambar 2.2.12 cardPosition

### 2.2.13 void printSolution(vector<string> hasil)

Fungsi ini akan dipanggil pada program utama untuk menampilkan semua solusi yang ditemukan di terminal.



```

1 void printSolution(vector<string> hasil)
2 // Menampilkan solusi
3 {
4     int number = 0;
5     cout << " -----
6     ----- " << endl;
7     cout << "|\\t" << solusi << " solusi ditemukan " <<
8     "\\t\\t\\t\\t\\t\\t\\t\\t|" << endl;
9     cout << " -----
10    ----- " << endl;
11    if (solusi == 0) {
12        cout << "|\\t" << "Tidak ada solusi
13        |" << endl;
14    }
15    else {
16        for (int i = 0; i < hasil.size(); i++) {
17            number++;
18            cout << " \\t" << number << ".\\t" << hasil[i] << endl;
19        }
20    }
21    cout << " -----
22    ----- " << endl;
23 }

```

Gambar 2.2.13 printSolution

### 2.2.14 void saveFile(vector<string> hasil)

Fungsi ini akan dipanggil pada program utama yang berfungsi untuk menyimpan hasil solusi jika pengguna menginginkannya.

```

1 void saveFile(vector<string> hasil)
2 // Menyimpan solusi ke file
3 {
4     ofstream file;
5     string namaFile, namaPath, pilihan;
6     int number = 0;
7
8     cout << " \t" << "Apakah ingin menyimpan solusi? (y/n) : ";
9     cin >> pilihan;
10    while (pilihan != "y" && pilihan != "n") {
11        cout << " " << endl;
12        cout << " \t" << "Input salah, masukkan kembali (y/n) : ";
13        cin >> pilihan;
14    }
15    if (pilihan == "y") {
16        cout << " " << endl;
17        namaPath = "../test/";
18        file.open(namaPath + "solusi_" + input1 + "_" + input2 + "_" + input3 + "_" + input4 + ".txt");
19        file << " -----" << endl;
20        file << "\t" << input1 << " " << input2 << " " << input3 << " " << input4 << endl;
21        file << " -----" << endl;
22        file << "\t" << solusi << " solusi ditemukan" << endl;
23        file << " -----" << endl;
24        if (solusi == 0) {
25            file << "\t" << "Tidak ada solusi" << endl;
26        }
27        else {
28            for (int i = 0; i < hasil.size(); i++) {
29                number++;
30                file << "\t" << number << ". \t" << hasil[i] << endl;
31            }
32        }
33        file << " -----" << endl;
34        file.close();
35        cout << " \t" << "File berhasil disimpan" << endl;
36        cout << " -----" << endl;
37    }
38    else if (pilihan == "n") {
39        cout << " " << endl;
40        cout << " \t" << "Tidak ada file yang disimpan" << endl;
41        cout << " -----" << endl;
42    }
43 }

```

Gambar 2.2.14 saveFile

## 2.3. Program Utama

Pada program utama akan dipanggil beberapa fungsi, yaitu intro() untuk menampilkan header sebagai pembuka, cardInput() untuk melakukan input secara random atau dari keyboard dan melakukan validasi, clock\_t start = clock() untuk menghitung waktu mulai program dalam menemukan solusi-solusi, cardPosition(cardA, cardB, cardC, cardD, &hasil) untuk mengecek apakah perhitungan 4 kartu dengan operator yang berbeda, penempatan kartu yang berbeda dan juga penempatan tanda kurung yang berbeda-beda akan mengeluarkan hasil 24. Jika iya, akan dimasukkan ke dalam vektor hasil, printSolution() untuk mengeprint semua solusi yang dihasilkan oleh 4 kartu, double end = clock() untuk menghitung waktu selesai program dalam

menemukan solusi-solusi, menampilkan waktu eksekusi program, dan saveFile(hasil) untuk menyimpan solusi-solusi yang dihasilkan.

A screenshot of a code editor with a dark background and light-colored text. The code is written in C++ and is enclosed in a main function. It includes a vector to store results, calls to intro(), cardInput(), cardPosition(), printSolution(), and saveFile(). It also uses clock() to measure execution time and cout to display it. The code is numbered from 1 to 19 on the left side.

```
1  int main()
2  {
3      vector<string> hasil;
4
5      intro();
6      cardInput();
7
8      clock_t start = clock();
9
10     cardPosition(card1, card2, card3, card4, &hasil);
11     printSolution(hasil);
12
13     double end = clock();
14     cout << " " << endl;
15     cout << " \t" << "Waktu eksekusi : " << (float)(clock() - start)/CLOCKS_PER_SEC << " detik" << endl;
16     cout << " " << endl;
17
18     saveFile(hasil);
19 }
```

Gambar 2.3.1 program utama

## BAB 3

### TEST CASE

#### 3.1 Test Case 1 (Random)

- Kartu : 2, 8, 2, 2
- Terdapat 3 kartu yang sama
- Melakukan input hingga benar
- Solusi ditemukan
- Terdapat waktu eksekusi program
- Berhasil menyimpan solusi

```

$$$$$  $$  $$          $$$$
$$ __$$ |$$ | $$ |     $$ __$$$
___/  $$ |$$ | $$ |     $$ /  ___| $$$$$$ $$$$$$ $$$$$$ $$$$$$&
$$$$$$ |$$$$$$$       $$ |$$$$  ___$$ |$$ __$  _$$ |  $$ __$|
$$ ____/  ____$$ |     $$ | _$$ | $$$$$$ |$$ /  $$ /  $$ |$$$$$$$ |
$$ |      $$ |     $$ |  $$ |$$ __$$ |$$ | $$ |  $$ |  $$ |  __|
$$$$$$$ |      $$ |     $$$$$$ |$$$$$$$ |$$ | $$ |  $$ |  $$$$$$
_____|      _|     ____/  ____|  _|  _|  _|  ____|

$$$$$      $$
$$ __$     $$ |
$$ /  ___| $$$$$$  $$ |$$  $$  $$$$$$  $$$$$$
$$$$$  $$ __$ |$$ |  $$  $$  |$$ __$ |$$ __$ |
___$  $$ /  $$ |$$ |  $$  $$ /  $$$$$$ |$$ |  _|
$$  $$ |$$ |  $$ |$$ |  $$$ /  $$ ____|$$ |
$$$$$ |$$$$$ |$$ |  $ /  $$$$$$  $$ |
_____/  ____/  _|  _/  ____|  _|

Pilih input dari keyboard atau random:
1. Keyboard
2. Random

Masukkan pilihan : 3

Input tidak valid. Masukkan pilihan (1/2): p

Input tidak valid. Masukkan pilihan (1/2): 2
```

```
-----
|      Kartu yang terpilih adalah 2, 8, 2, dan 2      |
|-----
|      22 solusi ditemukan      |
|-----
1.      ((2 + 8) + 2) * 2 = 24
2.      (2 + (8+ 2)) * 2 = 24
3.      2 * ((8+ 2) + 2) = 24
4.      2 * ((8- 2) * 2) = 24
5.      (2 * (8- 2)) * 2 = 24
6.      ((2 + 2) + 8) * 2 = 24
7.      (2 + (2+ 8)) * 2 = 24
8.      (2 + 2) * (8- 2) = 24
9.      2 * ((2+ 8) + 2) = 24
10.     ((2 * 2) + 8) * 2 = 24
11.     (2 * 2) * (8- 2) = 24
12.     (2 + (2/ 2)) * 8 = 24
13.     2 * ((2+ 2) + 8) = 24
14.     2 * ((2* 2) + 8) = 24
15.     ((2 / 2) + 2) * 8 = 24
16.     ((8 + 2) + 2) * 2 = 24
17.     (8 + (2+ 2)) * 2 = 24
18.     (8 + (2* 2)) * 2 = 24
19.     (8 - 2) * (2+ 2) = 24
20.     ((8 - 2) * 2) * 2 = 24
21.     (8 - 2) * (2* 2) = 24
22.     8 * ((2/ 2) + 2) = 24
-----

Waktu eksekusi : 0.015 detik

Apakah ingin menyimpan solusi? (y/n) : y

File berhasil disimpan
-----
```

Gambar 3.1.1 Test case 1

```

test > ≡ solusi_2_8_2_2.txt
1  -----
2      2 8 2 2
3  -----
4      22 solusi ditemukan
5  -----
6      1.      ((2 + 8) + 2) * 2 = 24
7      2.      (2 + (8+ 2)) * 2 = 24
8      3.      2 * ((8+ 2) + 2) = 24
9      4.      2 * ((8- 2) * 2) = 24
10     5.      (2 * (8- 2)) * 2 = 24
11     6.      ((2 + 2) + 8) * 2 = 24
12     7.      (2 + (2+ 8)) * 2 = 24
13     8.      (2 + 2) * (8- 2) = 24
14     9.      2 * ((2+ 8) + 2) = 24
15     10.     ((2 * 2) + 8) * 2 = 24
16     11.     (2 * 2) * (8- 2) = 24
17     12.     (2 + (2/ 2)) * 8 = 24
18     13.     2 * ((2+ 2) + 8) = 24
19     14.     2 * ((2* 2) + 8) = 24
20     15.     ((2 / 2) + 2) * 8 = 24
21     16.     ((8 + 2) + 2) * 2 = 24
22     17.     (8 + (2+ 2)) * 2 = 24
23     18.     (8 + (2* 2)) * 2 = 24
24     19.     (8 - 2) * (2+ 2) = 24
25     20.     ((8 - 2) * 2) * 2 = 24
26     21.     (8 - 2) * (2* 2) = 24
27     22.     8 * ((2/ 2) + 2) = 24
28     -----

```

Gambar 3.1.2 File solusi test case 1

### 3.2 Test Case 2 (Random)

- Kartu : K, 7, A, Q
- memiliki solusi
- Terdapat waktu eksekusi program
- Berhasil menyimpan solusi

```
Pilih input dari keyboard atau random:
1. Keyboard
2. Random

Masukkan pilihan : 2

-----
|      Kartu yang terpilih adalah K, 7, A, dan Q      |
|-----|
|      14 solusi ditemukan      |
|-----|
1.      ((13 + 1) / 7) * 12 = 24
2.      (13 + 1) / (7/ 12) = 24
3.      ((13 + 1) * 12) / 7 = 24
4.      (13 + 1) * (12/ 7) = 24
5.      ((1 + 13) / 7) * 12 = 24
6.      (1 + 13) / (7/ 12) = 24
7.      ((1 + 13) * 12) / 7 = 24
8.      (1 + 13) * (12/ 7) = 24
9.      12 * ((13+ 1) / 7) = 24
10.     (12 * (13+ 1)) / 7 = 24
11.     (12 / 7) * (13+ 1) = 24
12.     (12 / 7) * (1+ 13) = 24
13.     12 * ((1+ 13) / 7) = 24
14.     (12 * (1+ 13)) / 7 = 24
-----

Waktu eksekusi : 0.008 detik

Apakah ingin menyimpan solusi? (y/n) : y

File berhasil disimpan
-----
```

Gambar 3.2.1 Test case 2

```

test > ≡ solusi_K_7_A_Q.txt
1  -----
2      K 7 A Q
3  -----
4      14 solusi ditemukan
5  -----
6      1.      ((13 + 1) / 7) * 12 = 24
7      2.      (13 + 1) / (7/ 12) = 24
8      3.      ((13 + 1) * 12) / 7 = 24
9      4.      (13 + 1) * (12/ 7) = 24
10     5.      ((1 + 13) / 7) * 12 = 24
11     6.      (1 + 13) / (7/ 12) = 24
12     7.      ((1 + 13) * 12) / 7 = 24
13     8.      (1 + 13) * (12/ 7) = 24
14     9.      12 * ((13+ 1) / 7) = 24
15    10.      (12 * (13+ 1)) / 7 = 24
16    11.      (12 / 7) * (13+ 1) = 24
17    12.      (12 / 7) * (1+ 13) = 24
18    13.      12 * ((1+ 13) / 7) = 24
19    14.      (12 * (1+ 13)) / 7 = 24
20  -----

```

Gambar 3.2.2 Solusi test case 2

### 3.3 Test Case 3 (Random)

- Kartu : 7, J, Q, dan K
- Tidak memiliki solusi
- Melakukan input hingga benar
- Terdapat waktu eksekusi program
- Solusi berhasil disimpan





### 3.4 Test Case 4 (Keyboard)

- Kartu : 3, 8, 7, dan 9
- Memiliki solusi
- Terdapat waktu eksekusi program
- Berhasil menyimpan solusi

```
Pilih input dari keyboard atau random:
1. Keyboard
2. Random

Masukkan pilihan : 1

Masukkan 4 kartu:
(angka 2-10 atau huruf A, J, Q, dan K)

Masukkan kartu : 3 8 7 9

-----
|      Kartu yang anda pilih adalah 3, 8, 7, dan 9      |
|-----|
|      12 solusi ditemukan                             |
|-----|
1.      3 * ((7- 8) + 9) = 24
2.      3 * ((7+ 9) - 8) = 24
3.      3 * ((9- 8) + 7) = 24
4.      3 * ((9+ 7) - 8) = 24
5.      ((7 - 8) + 9) * 3 = 24
6.      (7 - (8- 9)) * 3 = 24
7.      ((7 + 9) - 8) * 3 = 24
8.      (7 + (9- 8)) * 3 = 24
9.      ((9 - 8) + 7) * 3 = 24
10.     (9 - (8- 7)) * 3 = 24
11.     ((9 + 7) - 8) * 3 = 24
12.     (9 + (7- 8)) * 3 = 24
-----

Waktu eksekusi : 0.018 detik

Apakah ingin menyimpan solusi? (y/n) : y

File berhasil disimpan
-----
```

Gambar 3.4.1 Test case 4

```

test > ≡ solusi_3_8_7_9.txt
1  -----
2      3 8 7 9
3  -----
4      12 solusi ditemukan
5  -----
6      1.      3 * ((7- 8) + 9) = 24
7      2.      3 * ((7+ 9) - 8) = 24
8      3.      3 * ((9- 8) + 7) = 24
9      4.      3 * ((9+ 7) - 8) = 24
10     5.      ((7 - 8) + 9) * 3 = 24
11     6.      (7 - (8- 9)) * 3 = 24
12     7.      ((7 + 9) - 8) * 3 = 24
13     8.      (7 + (9- 8)) * 3 = 24
14     9.      ((9 - 8) + 7) * 3 = 24
15    10.      (9 - (8- 7)) * 3 = 24
16    11.      ((9 + 7) - 8) * 3 = 24
17    12.      (9 + (7- 8)) * 3 = 24
18  -----

```

Gambar 3.4.2 Solusi test case 4

### 3.5 Test Case 5 (Keyboard)

- Kartu : A, 7, 4, dan A
- Terdapat 2 kartu yang sama
- Solusi ditemukan
- Terdapat waktu eksekusi program
- Berhasil menyimpan hasil solusi

```

Pilih input dari keyboard atau random:
1. Keyboard
2. Random

Masukkan pilihan : 1

Masukkan 4 kartu:
(angka 2-10 atau huruf A, J, Q, dan K)

Masukkan kartu : A 7 4 A

```

```
-----
|      Kartu yang anda pilih adalah A, 7, 4, dan A      |
|-----
|      29 solusi ditemukan                               |
|-----
1.      (1 + 7) * (4- 1) = 24
2.      ((1 * 7) - 1) * 4 = 24
3.      1 * ((7- 1) * 4) = 24
4.      (1 * (7- 1)) * 4 = 24
5.      (1 * 4) * (7- 1) = 24
6.      (7 + 1) * (4- 1) = 24
7.      ((7 - 1) * 4) * 1 = 24
8.      (7 - 1) * (4* 1) = 24
9.      ((7 - 1) * 4) / 1 = 24
10.     (7 - 1) * (4/ 1) = 24
11.     ((7 - 1) * 1) * 4 = 24
12.     (7 - 1) * (1* 4) = 24
13.     (7 - (1* 1)) * 4 = 24
14.     ((7 - 1) / 1) * 4 = 24
15.     (7 - (1/ 1)) * 4 = 24
16.     (7 - 1) / (1/ 4) = 24
17.     ((7 * 1) - 1) * 4 = 24
18.     ((7 / 1) - 1) * 4 = 24
19.     (4 - 1) * (7+ 1) = 24
20.     (4 * 1) * (7- 1) = 24
21.     4 * ((1* 7) - 1) = 24
22.     (4 / 1) * (7- 1) = 24
23.     (4 - 1) * (1+ 7) = 24
24.     4 * ((7- 1) * 1) = 24
25.     (4 * (7- 1)) * 1 = 24
26.     4 * ((7- 1) / 1) = 24
27.     (4 * (7- 1)) / 1 = 24
28.     4 * ((7* 1) - 1) = 24
29.     4 * ((7/ 1) - 1) = 24
-----

Waktu eksekusi : 0.034 detik

Apakah ingin menyimpan solusi? (y/n) : y

File berhasil disimpan
-----
```

Gambar 3.5.1 Test case 5

```

test > ≡ solusi_A_7_4_A.txt
1  -----
2      A 7 4 A
3  -----
4      29 solusi ditemukan
5  -----
6      1.      (1 + 7) * (4- 1) = 24
7      2.      ((1 * 7) - 1) * 4 = 24
8      3.      1 * ((7- 1) * 4) = 24
9      4.      (1 * (7- 1)) * 4 = 24
10     5.      (1 * 4) * (7- 1) = 24
11     6.      (7 + 1) * (4- 1) = 24
12     7.      ((7 - 1) * 4) * 1 = 24
13     8.      (7 - 1) * (4* 1) = 24
14     9.      ((7 - 1) * 4) / 1 = 24
15     10.     (7 - 1) * (4/ 1) = 24
16     11.     ((7 - 1) * 1) * 4 = 24
17     12.     (7 - 1) * (1* 4) = 24
18     13.     (7 - (1* 1)) * 4 = 24
19     14.     ((7 - 1) / 1) * 4 = 24
20     15.     (7 - (1/ 1)) * 4 = 24
21     16.     (7 - 1) / (1/ 4) = 24
22     17.     ((7 * 1) - 1) * 4 = 24
23     18.     ((7 / 1) - 1) * 4 = 24
24     19.     (4 - 1) * (7+ 1) = 24
25     20.     (4 * 1) * (7- 1) = 24
26     21.     4 * ((1* 7) - 1) = 24
27     22.     (4 / 1) * (7- 1) = 24
28     23.     (4 - 1) * (1+ 7) = 24
29     24.     4 * ((7- 1) * 1) = 24
30     25.     (4 * (7- 1)) * 1 = 24
31     26.     4 * ((7- 1) / 1) = 24
32     27.     (4 * (7- 1)) / 1 = 24
33     28.     4 * ((7* 1) - 1) = 24
34     29.     4 * ((7/ 1) - 1) = 24
35  -----

```

Gambar 3.5.2 Solusi test case 5

### 3.6 Test Case 6 (Keyboard)

- Kartu : 3, 3, 3, dan 3
- Memiliki 4 kartu yang sama
- Terdapat waktu eksekusi program
- Berhasil simpan jawaban

```
Pilih input dari keyboard atau random:
1. Keyboard
2. Random

Masukkan pilihan : 1

Masukkan 4 kartu:
(angka 2-10 atau huruf A, J, Q, dan K)

Masukkan kartu : 3 3 3 3

-----
|      Kartu yang anda pilih adalah 3, 3, 3, dan 3      |
|-----|
|      2 solusi ditemukan                               |
|-----|
1.      ((3 * 3) * 3) - 3 = 24
2.      (3 * (3* 3)) - 3 = 24
|-----|

Waktu eksekusi : 0.004 detik

Apakah ingin menyimpan solusi? (y/n) : y

File berhasil disimpan
-----
```

Gambar 3.6.1 Test case 6

```
test > ≡ solusi_3_3_3_3.txt
1  -----
2      3 3 3 3
3  -----
4      2 solusi ditemukan
5  -----
6      1.      ((3 * 3) * 3) - 3 = 24
7      2.      (3 * (3* 3)) - 3 = 24
8  -----
```

Gambar 3.6.2 Solusi test case 6

### 3.7 Test Case 7 (Keyboard)

- Kartu : 3, 4, 4, dan 4
- Terdapat 3 kartu yang sama
- Melakukan input kartu hingga benar
- Solusi ditemukan
- Terdapat waktu eksekusi program
- Berhasil menyimpan hasil solusi

```
Pilih input dari keyboard atau random:
1. Keyboard
2. Random

Masukkan pilihan : h

Input tidak valid. Masukkan pilihan (1/2): 1

Masukkan 4 kartu:
(angka 2-10 atau huruf A, J, Q, dan K)

Masukkan kartu : 99 y 4 3

Kartu tidak valid. Masukkan kembali 4 kartu.
(angka 2-10 atau huruf A, J, Q, dan K)

Masukkan 4 kartu: 1 1 1 22

Kartu tidak valid. Masukkan kembali 4 kartu.
(angka 2-10 atau huruf A, J, Q, dan K)

Masukkan 4 kartu: 3 4 4 4

-----
|      Kartu yang anda pilih adalah 3, 4, 4, dan 4      |
|-----|
|      4 solusi ditemukan                               |
|-----|
1.      ((3 + 4) * 4) - 4 = 24
2.      ((4 + 3) * 4) - 4 = 24
3.      (4 * (3+ 4)) - 4 = 24
4.      (4 * (4+ 3)) - 4 = 24
|-----|

Waktu eksekusi : 0.007 detik

Apakah ingin menyimpan solusi? (y/n) : p

Input salah, masukkan kembali (y/n) : y

File berhasil disimpan
-----
```

Gambar 3.7.1 Test case 7

```
test > ≡ solusi_3_4_4_4.txt
1 -----
2      3 4 4 4
3 -----
4      4 solusi ditemukan
5 -----
6      1.      ((3 + 4) * 4) - 4 = 24
7      2.      ((4 + 3) * 4) - 4 = 24
8      3.      (4 * (3+ 4)) - 4 = 24
9      4.      (4 * (4+ 3)) - 4 = 24
10 -----
```

Gambar 3.7.2 Solusi test case 7

### 3.8 Test case 8 (Keyboard)

- Kartu : J, Q, K, dan 2
- Solusi ditemukan
- Terdapat waktu eksekusi program
- Solusi berhasil disimpan



```
Pilih input dari keyboard atau random:
1. Keyboard
2. Random

Masukkan pilihan : 1

Masukkan 4 kartu:
(angka 2-10 atau huruf A, J, Q, dan K)

Masukkan kartu : J Q K 2

-----
|      Kartu yang anda pilih adalah J, Q, K, dan 2      |
|-----|
|      16 solusi ditemukan                               |
|-----|
1.      ((11 - 12) + 13) * 2 = 24
2.      (11 - (12- 13)) * 2 = 24
3.      ((11 + 13) - 12) * 2 = 24
4.      (11 + (13- 12)) * 2 = 24
5.      ((11 + 13) / 2) + 12 = 24
6.      12 + ((11+ 13) / 2) = 24
7.      12 + ((13+ 11) / 2) = 24
8.      ((13 + 11) - 12) * 2 = 24
9.      (13 + (11- 12)) * 2 = 24
10.     ((13 + 11) / 2) + 12 = 24
11.     ((13 - 12) + 11) * 2 = 24
12.     (13 - (12- 11)) * 2 = 24
13.     2 * ((11- 12) + 13) = 24
14.     2 * ((11+ 13) - 12) = 24
15.     2 * ((13+ 11) - 12) = 24
16.     2 * ((13- 12) + 11) = 24

-----

Waktu eksekusi : 0.01 detik

Apakah ingin menyimpan solusi? (y/n) : y

File berhasil disimpan

-----
```

Gambar 3.8.1 Test case 8

### 3.9 Test Case 9 (Keyboard)

- Kartu : 8, 10, A, dan 3
- Solusi ditemukan
- Terdapat waktu eksekusi program
- Solusi berhasil disimpan

```
Pilih input dari keyboard atau random:
1. Keyboard
2. Random

Masukkan pilihan : 1

Masukkan 4 kartu:
(angka 2-10 atau huruf A, J, Q, dan K)

Masukkan kartu : 8 10 A 3

-----
|      Kartu yang anda pilih adalah 8, 10, A, dan 3      |
|-----|
|      7 solusi ditemukan                                |
|-----|
1.      8 * ((10- 1) / 3) = 24
2.      (8 * (10- 1)) / 3 = 24
3.      (8 / 3) * (10- 1) = 24
4.      ((10 - 1) * 8) / 3 = 24
5.      (10 - 1) * (8/ 3) = 24
6.      ((10 - 1) / 3) * 8 = 24
7.      (10 - 1) / (3/ 8) = 24
|-----|

Waktu eksekusi : 0.008 detik

Apakah ingin menyimpan solusi? (y/n) : y

File berhasil disimpan
|-----|
```

Gambar 3.9.1 Test case 9

## BAB 4

### LAMPIRAN

#### 4.1 Link Repository

Link : [https://github.com/lostgirlll/Tucil1\\_13521030](https://github.com/lostgirlll/Tucil1_13521030)

#### 4.2 Cek List

Poin	Ya	Tidak
1. Program berhasil dikompilasi tanpa kesalahan	✓	
2. Program berhasil running	✓	
3. Program dapat membaca input / generate sendiri dan memberikan luaran	✓	
4. Solusi yang diberikan program memenuhi (berhasil mencapai 24)	✓	
5. Program dapat menyimpan solusi dalam file teks	✓	